Claims

1. A method for creating the data structure of a programmable state machine to parse an input word chain by identifying a word pattern within said input word chain to point to a resulting address, said method comprising the steps of:

- creating a state table corresponding to states of said programmable state machine for identifying the word pattern in the input word chain, each state table entry comprising a s-bit current state, an input n-bit word and a s-bit next state;
- reducing the number of entries in the state table by converting the entries into a reduced number of state-transition rule entries, each containing a ternary match condition expressed as a test value comprising the s-bit current state, the input n-bit word, a test mask on the s-bit current state and the input n-bit word in combination, and the s-bit next state; and,
- ordering the reduced state table entries obtained by the execution of the preceding step, in a prioritized order, with most frequently used transition rules having the highest priority.

2. The method of claim 1 further comprising the steps of:

- defining as a hash index, for the reduced state table, a set of i bit locations inside the s-bit current state and the input n-bit word in combination, and an integer N, such that, at most, N table entries can match a hash index value;
- creating a compressed state table, indexed by the hash index, having 2^i entries, each entry corresponding to one value of the hash index, and each having a maximum of N transition rules of the reduced state table corresponding to

the same hash index value and written in a priority order; and,

- saving an s + n bit index mask corresponding to the hash index, said index mask having bits coded to a first binary value except for the hash index bit locations coded to a second binary value and saving a base address pointer (SP1, SP2) of the compressed space table.

3. The method of claim 1 or 2, further comprising the step of :

- dividing the compressed state-transition rule table into more than one compressed state-transition rule table; and,
- extending in each of the divided compressed state tables, each state-transition rule with the index mask and a base address pointer of the divided compressed table of the next state in said state-transition rule.

4. The method for parsing an input word chain using a data structure of a programmable state machine created according to anyone of claims 1 to 3, said method comprising the steps of:

- initializing the current state, a current index bit mask of the data structure and a current base pointer;
- defining the first word of the input word chain as being the current input;
- extracting the hash index from the current state and current input according to the index mask;
- searching in the space table indicated by the current base pointer, the entry corresponding to the hash index and searching for the state-transition rule matching the current state current input, if multiple transition rules match, selecting one with the highest priority;
- if the next state is not a final state, extracting the new values for the current state, the current index mask of the data structure and the current base pointer;
- defining the next word of the input word chain as being the current input;

- repeating the preceding extracting hash index, searching and extracting new values until the next read state is final.

5. The method for searching a resulting value corresponding to an input word chain using a data structure of a programmable state machine created according to anyone of claims 1 to 3 wherein the compressed tables have been built using a hash index taken among the n bits of the word input, the two left-most bits within the test vectors, that correspond to the state register, are unused, the next state bits of each transition rule are unused and the base address pointer of the transition rules may include a final result (P, R, Q), said method comprising the steps of:
- defining the first word of the input word chain as being the current input;
- extracting the hash index from the current input according to the index mask;
- searching in the space table indicated by the current base pointer, the entry corresponding to the hash index and searching for the state-transition rule matching the current state current input, if multiple transition rules match, selecting one with the highest priority;
- reading the base address pointer field, and, if it does not include a final result, repeating the following steps until the base address pointer field includes the final result;
- defining the next word of the input word chain as being the current input;
- checking that the next word maps the test mask of the transition rule pointed by the read base address pointer;
- reading the base address pointer field.

6. A method for performing wire-speed deep packet processing comprising the steps of, upon reception of an input packet consisting in a variable chain of words, repeating alternatively the steps of the method of claim 4

and the steps of the method of claim 5 until all the words have been processed.

7. A method for performing wire-speed deep packet processing comprising the steps of, upon reception of an input packet consisting in a variable chain of words, combining in an order corresponding to the deep packet processing the set of steps of the method of claim 4 with the set of steps of the method of claim 5 until all the words have been processed.

8. An apparatus for deep packet processing comprising means adapted for implementing the steps of the method according to anyone of claims 1 to 7.

9. A chip embedded apparatus comprising means adapted for implementing the steps of the method according to anyone of claims 1 to 7.

10. A computer program product comprising programming code instructions for executing the steps of the method according to anyone of claims 1 to 7 when said program is executed on a computer.